AMENDMENTS TO THE CLAIMS

The claims have been amended as follows:

19-30. (Canceled)

31. (Currently Amended) A transaction based constraint enforcer for a database system, for enforcing a set of constraints that governs the integrity of information stored in the database system, said enforcer being arranged to delay constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction, the constraint enforcer comprising

a stack maker module, arranged for creating and updating said check stack, said stack maker module being operatively connected to a runtime module in the database system and arranged to receive data from said runtime module,

wherein the check stack contains a list of functions that have to be executed at the end of the transaction, said functions originating from Insert, Delete and Update Data Manipulation Language (DML) operations calling up the stack maker module,

the Insert DML operation calling up the stack maker module leading to an insert process being performed on the check stack,

the insert process involving placing all checks that have to be executed as a result of an occurrence of a table type being inserted and corresponding conceptual rules being identified for the table type being inserted,

the Delete DML operation calling up the stack maker module leading to a delete process being performed on the check stack,

the delete process involving removing previously inserted entries on the check stack for the occurrence to be deleted and placing all checks that have to be executed as a result of a table type being deleted and corresponding conceptual

rules being identified for the table type being deleted, and

the Update DML operation calling up the stack maker module leading to said delete process followed by said insert process being performed on the check stack,

an enforcer module, arranged to receive check data from the check stack, to process the check data received from the check stack, and to provide resulting data to the runtime module,

a conceptual rules module wherein said constraints are stored in the form of rules for prescribing permitted states and transitions that the database can undertake, the conceptual rules module being operatively connected to said stack maker module,

said stack maker module being arranged to retrieve constraints from said conceptual rules module,

wherein said constraints are constraints executed within the transaction which allow conceptual rules to be broken during the transaction, but allow the database system to be in a consistent state at the beginning and end of the transaction.

32. (Previously Presented) Constraint enforcer according to claim 31, wherein said check stack is stored on persistent or volatile memory.

33. (Canceled)

34. (Canceled)

35. (Previously Presented) Constraint enforcer according to claim 31, wherein said constraints are selected from: primary keys, foreign keys, subset constraints, and exclude constraints.

36. (Currently Amended) Method for enforcing a set of constraints that governs the integrity of information stored in a database system, the constraints being stored in a conceptual rules module in the form of rules for prescribing permitted states and transitions that the database can undertake, the

method comprising the steps of

delaying constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction,

by a stack maker module operatively connected to a runtime module in said database system: receiving data from said runtime module, and

creating and updating said check stack, and retrieving constraints from said conceptual rules module, ~~and~~

wherein the check stack contains a list of functions that have to be executed at the end of the transaction, said functions originating from Insert, Delete and Update Data Manipulation Language (DML) operations calling up the stack maker module,

the Insert DML operation calling up the stack maker module leading to an insert process being performed on the check stack,

the insert process involving placing all checks that have to be executed as a result of an occurrence of a table type being inserted and corresponding conceptual rules being identified for the table type being inserted,

the Delete DML operation calling up the stack maker module leading to a delete process being performed on the check stack,

the delete process involving removing previously inserted entries on the check stack for the occurrence to be deleted and placing all checks that have to be executed as a result of a table type being deleted and corresponding conceptual rules being identified for the table type being deleted, and

the Update DML operation calling up the stack maker module leading to said delete process followed by said insert process being performed on the check stack, and

by an enforcer module: receiving check data from the check stack, processing the check data received from the check stack, and providing resulting data to the runtime module,

wherein said constraints are constraints executed within the transaction which allow conceptual rules to be broken during the transaction, but allow the database system to be in a consistent state at the beginning and end of the transaction.

37. (Previously Presented) Method according to claim 36, wherein said check stack is stored on persistent or volatile memory.

38. (Canceled)

39. (Canceled)

40. (Previously Presented) Method according to claim 36, wherein said constraints are selected from: primary keys, foreign keys, subset constraints, and exclude constraints.

41. (Currently Amended) A database system, comprising an application program interface, providing a two-way message interface to a user application program,

a runtime module, operatively connected to the application program interface, a storage engine module, operatively connected to the runtime module, a data storage, operatively connected to the storage engine module, and

a transaction based constraint enforcer, for enforcing a set of constraints that governs the integrity of information stored in the database system, said enforcer being arranged to delay constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction, said constraint enforcer further comprising

a stack maker module, arranged for creating and updating said check stack, said stack maker module being operatively connected to a runtime module in the database system and arranged to receive data from said runtime module,

wherein the check stack contains a list of functions that have to be executed at the end of the transaction, said functions originating from Insert, Delete and Update Data Manipulation

Language (DML) operations calling up the stack maker module,

the Insert DML operation calling up the stack maker module leading to an insert process being performed on the check stack,

the insert process involving placing all checks that have to be executed as a result of an occurrence of a table type being inserted and corresponding conceptual rules being identified for the table type being inserted,

the Delete DML operation calling up the stack maker module leading to a delete process being performed on the check stack,

the delete process involving removing previously inserted entries on the check stack for the occurrence to be deleted and placing all checks that have to be executed as a result of a table type being deleted and corresponding conceptual rules being identified for the table type being deleted, and

the Update DML operation calling up the stack maker module leading to said delete process followed by said insert process being performed on the check stack,

an enforcer module, arranged to receive check data from the check stack, to process the check data received from the check stack, and to provide resulting data to the runtime module,

a conceptual rules module wherein said constraints are stored in the form of rules for prescribing permitted states and transitions that the database can undertake, the conceptual rules module being operatively connected to said stack maker module,

said stack maker module being arranged to retrieve constraints from said conceptual rules module,

wherein said constraints are constraints executed within the transaction which allow conceptual rules to be broken during the transaction, but allow the database system to be in a consistent state at the beginning and end of the transaction.

42. (Previously Presented) System according to claim 41, wherein said check stack is stored on persistent or volatile memory.

43. (Canceled)

44. (Canceled)

45. (Previously Presented) System according to claim 41, wherein said constraints are selected from: primary keys, foreign keys, subset constraints, and exclude constraints.

46. (New) Constraint enforcer according to claim 31, wherein the enforcer module is further arranged to inform a caller with a message retrieved from the check stack entries associated with detected integrity violations.

47. (New) Method according to claim 36, further comprising the step of informing a caller with a message retrieved from the check stack entries associated with detected integrity violations.

48. (New) System according to claim 41, wherein the enforcer module is further arranged to inform a caller with a message retrieved from the check stack entries associated with detected integrity violations.